(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7302223号
(P7302223)

(45)発行日　令和5年7月4日(2023.7.4)

(24)登録日　令和5年6月26日(2023.6.26)

(51)国際特許分類

Ｇ０６Ｆ　11/34　(2006.01)		Ｇ０６Ｆ	11/34	１４７
Ｇ０６Ｆ　11/07　(2006.01)		Ｇ０６Ｆ	11/07	１９０
Ｇ０６Ｆ　21/55　(2013.01)		Ｇ０６Ｆ	11/07	１４０Ｅ
		Ｇ０６Ｆ	21/55	３４０

ＦＩ

請求項の数　10　（全26頁）

(21)出願番号　特願2019-58148(P2019-58148)
(22)出願日　平成31年3月26日(2019.3.26)
(65)公開番号　特開2020-160679(P2020-160679A)
(43)公開日　令和2年10月1日(2020.10.1)
審査請求日　令和4年2月3日(2022.2.3)

(73)特許権者　000004237
日本電気株式会社
東京都港区芝五丁目７番１号
(74)代理人　100080816
弁理士　加藤 朝道
(74)代理人　100098648
弁理士　内田 潔人
(72)発明者　加藤 剛史
東京都港区芝五丁目７番１号 日本電気株式会社内
審査官　渡辺 一帆

最終頁に続く

(54)【発明の名称】　スクリプト検出装置、方法及びプログラム

(57)【特許請求の範囲】

【請求項１】

監視対象システム上のプロセスの振る舞いの監視の結果収集されたイベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成する第１の手段と、

分析対象のプログラムに関するモデルについて類似度に基づきグループ化できる場合、前記分析対象のプログラムはスクリプトを読み込んで実行するインタプリタであると判断しスクリプトの特定を行う第２の手段と、

を含む、スクリプト検出装置。

【請求項２】

前記第２の手段で特定されたスクリプトが記憶部に格納されているスクリプトであるか否かを判別し、前記特定されたスクリプトが既知のプロセスであるか否かを判別する第３の手段をさらに含む、請求項１に記載のスクリプト検出装置。

【請求項３】

前記第１乃至第３の手段を備えたスクリプト検出部を備え、さらに、

前記第１の手段と同様に、前記イベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成するモデル作成部と、

前記第２の手段と同様にして、スクリプトを特定し、特定したスクリプトの情報を前記記憶部に格納するモデル分析部と、

前記記憶部と、

を含む、請求項２に記載のスクリプト検出装置。

【請求項４】
前記第２の手段は、類似したモデルでグループ化できた場合、前記インタプリタの共通の振る舞いとして、前記インタプリタによる操作と、操作対象を関連付けて前記記憶部に記憶し、
さらに、前記スクリプトの共通の振る舞いとして、前記スクリプトと、前記スクリプトを実行するインタプリタによる操作、操作対象を関連付けて前記記憶部に記憶する、請求項２又は３に記載のスクリプト検出装置。
【請求項５】
前記第３の手段は、前記第１の手段で組み立てた前記モデルに関して、前記スクリプトの振る舞いが、前記記憶部から取得した前記スクリプトの共通の振る舞いを包含している場合には、前記インタプリタである前記分析対象のプログラムにより前記特定したスクリプトファイルが実行されたと判断し、前記スクリプトの振る舞いが、前記記憶部から取得した前記スクリプトの共通の振る舞いを包含しない場合には、前記分析対象のプログラムは、既知のスクリプトを実行していないと判断する、請求項４に記載のスクリプト検出装置。
【請求項６】
前記第２の手段は、類似度に基づきグループ化を行うにあたり、スクリプトの相違によらずインタプリタで共通に行われる振る舞いを外した上で前記グループ化を行う、請求項１乃至５のいずれか１項に記載のスクリプト検出装置。
【請求項７】
前記第２の手段は、子プロセスの振る舞いを、親プロセスである大元のプログラムの振る舞いに埋め込み、前記大元のプログラムの時間軸上の振る舞いとして展開したモデルを生成する、請求項１乃至６のいずれか１項に記載のスクリプト検出装置。
【請求項８】
請求項１乃至７のいずれか１項に記載のスクリプト検出装置と、
前記スクリプト検出装置に通信接続され、前記スクリプト検出装置からスクリプト検出結果を受け取り、前記スクリプト検出結果を利用して前記監視対象システムに関する分析を行うスクリプト情報利用装置と、
を備えた管理システム。
【請求項９】
コンピュータによって実行されるスクリプト検出方法であって、
プロセスの振る舞いの監視の結果収集されたイベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成し、
分析対象のプログラムに関するモデルについて類似度に基づきグループ化できる場合、前記分析対象のプログラムはスクリプトを読み込んで実行するインタプリタであると判断しスクリプトの特定を行う、スクリプト検出方法。
【請求項１０】
プロセスの振る舞いの監視の結果収集されたイベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成する処理と、
分析対象のプログラムに関するモデルについて類似度に基づきグループ化できる場合、前記分析対象のプログラムはスクリプトを読み込んで実行するインタプリタであると判断しスクリプトの特定を行う処理と、
をコンピュータに実行させるプログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、スクリプト検出装置、方法及びプログラムに関する。
【背景技術】
【０００２】
システムやプロセス等の振る舞い（挙動）からサイバー攻撃を検知するシステム技術が

実用化されている。
【０００３】
例えば特許文献１には、プログラムが監視されるべきであると決定される場合、前記プログラムに関連する複数のイベントをトレースし、前記トレースが終了する前にトレースされるべきイベントの数を決定し、前記プロセスがマルウェアを含むか否かの判断に対して複数の前記トレースされたイベントの複数の結果を解析する装置が開示されている。
【０００４】
また特許文献２には、対象コンピュータにおける不正な挙動を検出し、その挙動を起こすソースコード又はオブジェクトコードの少なくともいずれかのコードを抽出する構成が開示されている。
【０００５】
さらに、正常時のプロセスの振る舞いを機械学習し、正常時との違いから異常を検知するシステム等も知られている（例えば非特許文献１）。
【先行技術文献】
【特許文献】
【０００６】
【文献】特表２０１７－５２２６４１号公報
特開２０１０－１９８０５４号公報
【非特許文献】
【０００７】
【文献】多賀戸裕樹、栄純明、喜田弘司、朝倉敬喜、“未知のサイバー攻撃を自動検知する自己学習型システム異常検知技術（ASI）”、NEC技報／Vol.69 No.1（2016年９月）
【発明の概要】
【発明が解決しようとする課題】
【０００８】
以下に関連技術の分析を与える。
【０００９】
プロセスの振る舞いからマルウェアなどによるサイバー攻撃を検知する関連技術のシステムにおいて、マルウェアによる攻撃を検知しても、例えば、スクリプトとして書かれたマルウェア本体のパスが実行コマンドの引数として渡されていない場合や、スクリプト自体が難読化されている場合がある。このような場合、マルウェア本体を特定できないという事態も生じ得る。
【００１０】
また、プロセスの振る舞いからマルウェアなどによるサイバー攻撃を検知する関連技術のシステムにおいて、スクリプトを使った攻撃をプロセスの振る舞いとして観察していると、インタプリタが攻撃しているように見えてしまうという問題もある。例えばWindows（登録商標） PowerShell（マイクロソフトが開発した拡張可能なCommand Line Interface（CLI) シェル及びスクリプト言語）の実行ファイルpowershell.exeが不正な振る舞いを行ったとしても、それ自体は正規のプログラムであり、powershell.exeに不正な振る舞いをさせているのは、powershell.exeで動作するスクリプトである。
【００１１】
攻撃内容が難読化されているケースでは、スクリプトをプロセスの引数などのメタ情報から特定することが難しい。このため、攻撃に使われているスクリプトを機械的に特定することは困難である。
【００１２】
また、正常時のプロセスの振る舞いを学習し、正常時との違いから異常を検知する関連技術（非特許文献１等）のシステムにおいて、実行中のスクリプトの違いを認識させずに、インタプリタの振る舞いを学習させると、平常時に多様な振る舞いをするプログラムとして学習されてしまう。その結果、異常を検知する機会が失われる。
【００１３】

正常時のプロセスの振る舞いを学習し、正常時との違いから異常を検知する関連技術（非特許文献１等）のシステムにおいて、あるプログラムがインタプリタであることを予めシステムへ教えておくことで学習精度を向上できる可能性はある。

【００１４】

しかし、入力によって振る舞いが変わり得るプログラムは、インタプリタとして攻撃に使われる可能性がある。したがって、これらのプログラムを実環境からすべて特定することは困難である。

【００１５】

一般的なＯＳ（Operating System）では、プロセスの振る舞いを、システムコールなどの単位で外部プロセスから監視する手段が提供される。しかし、スクリプトを実行するインタプリタがスクリプトの実態（ファイルなど）に対して行う振る舞いは読み込み（ｒｅａｄ）のみであり、この振る舞いから、インタプリタとスクリプトを機械的に特定することはできない。

【００１６】

なお、上記した特許文献１、２には、対象ファイルがスクリプトであることが分かっているということを前提とした発明は開示されているが、実行されたプログラムの振る舞いを分析してインタプリタとマルウェアを探し出す構成は開示も示唆もされていない。

【００１７】

本発明は、上記事情に鑑みて創案されたものであって、その目的の一つは、プロセスの振る舞いからインタプリタとスクリプトを機械的に特定可能とする装置、方法、プログラムを提供することにある。

【課題を解決するための手段】

【００１８】

本発明のいくつかの形態の一つによれば、監視対象システム上のプロセスの振る舞いの監視の結果収集されたイベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成する第１の手段と、分析対象のプログラムに関するモデルについて類似度に基づきグループ化できる場合、前記分析対象のプログラムはスクリプトを読み込んで実行するインタプリタであると判断しスクリプトの特定を行う第２の手段と、を含むスクリプト検出装置が提供される。

【００１９】

本発明の別の一つの形態によれば、プロセスの振る舞いの監視の結果収集されたイベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成し、

分析対象のプログラムに関するモデルについて類似度に基づきグループ化できる場合、前記分析対象のプログラムはスクリプトを読み込んで実行するインタプリタであると判断しスクリプトの特定を行うスクリプト検出方法が提供される。

【００２０】

本発明のさらに別の一つの形態によれば、プロセスの振る舞いの監視の結果収集されたイベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成する処理と、

分析対象のプログラムに関するモデルについて類似度に基づきグループ化できる場合、前記分析対象のプログラムはスクリプトを読み込んで実行するインタプリタであると判断しスクリプトの特定を行う処理と、

をコンピュータに実行させるプログラムが提供される。

【００２１】

本発明のさらに他の形態によれば、上記プログラムを記憶したコンピュータ可読記録媒体（computer readable recording medium）が提供される。この記録媒体は、ＲＡＭ（Random Access Memory）、ＲＯＭ（Read Only Memory)、又は、EEPROM(Electrically Erasable and Programmable ROM））等の半導体ストレージや、ＨＤＤ（Hard Disk Drive）、ＳＳＤ（Solid State Drive）、ＣＤ（Compact Disc）、又は、ＤＶＤ（Digital Versatile Disc）等の非一時的コンピュータ可読媒体（non-transitory c

omputer readable medium）からなる。
【発明の効果】
【００２２】
本発明によれば、プロセスの振る舞いからインタプリタとスクリプトを機械的に特定可能としている。
【図面の簡単な説明】
【００２３】
【図１】本発明の例示的な実施形態の構成を例示する図である。
【図２】本発明の例示的な実施形態を説明する図である。
【図３】本発明の例示的な実施形態を説明する図である。
【図４】本発明の例示的な実施形態を説明する図である。
【図５】本発明の例示的な実施形態のモデルきおくを説明する図である。
【図６】本発明の例示的な実施形態を説明する図である。
【図７】本発明の例示的な実施形態を説明する図である。
【図８】本発明の例示的な実施形態におけるモデル分析部の動作の一例を説明する流れ図である。
【図９】本発明の例示的な実施形態を説明する図である。
【図１０】本発明の例示的な実施形態を説明する図である。
【図１１】本発明の例示的な実施形態を説明する図である。
【図１２】本発明の例示的な実施形態を説明する図である。
【図１３】（Ａ）、（Ｂ）は本発明の例示的な実施形態を説明する図である。
【図１４】本発明の例示的な実施形態におけるスクリプト検出部の動作の一例を説明する図である。
【図１５】本発明の例示的な実施形態（コンピュータ装置の実装例）を説明する図である。
【図１６】本発明の例示的な実施形態のプログラムを説明する図である。
【発明を実施するための形態】
【００２４】
本発明の一実施形態によれば、スクリプト検出装置は、監視対象システムで実行されるプロセスの振る舞いを収集し、繰り返し実行されるプログラムの振る舞いを動的に分析し、インタプリタとして動作するプログラムとスクリプトを検出する。すなわち、インタプリタとして動作するプログラムの振る舞いを動的に分析することで、インタプリタとスクリプトを機械的に特定可能としている。
【００２５】
さらに、本発明の一実施形態によれば、インタプリタとスクリプトの振る舞いを学習することで、新しく導入（実行）されたスクリプトであるかを特定可能としている。
【００２６】
本発明の一実施形態では、インタプリタが以下のように動作する可能性があることを想定して分析を行う。
【００２７】
・子プロセスを多用し、目的の処理を子プロセスに実行させる。
【００２８】
また、本発明の一実施形態では、インタプリタには、以下の特徴があることを利用して分析を行う。
【００２９】
・インタプリタが同じスクリプトを実行しているならば、その振る舞いは類似するはずである。
・インタプリタはスクリプトを実行する前にスクリプトを読み込むはずである。
【００３０】
以下、本発明の例示的な実施形態について図面を参照して説明する。図１は、本発明の例示的な実施形態のシステム構成を模式的に例示する図である。

【００３１】
スクリプト検出装置２０は、監視対象システム１０から受信したイベント情報を分析する。スクリプト検出装置２０は、イベント情報に基づき、プロセスの振る舞いのモデルを組み立てて分析し、インタプリタとスクリプトを検出してスクリプト情報利用装置３０へ送信する。
【００３２】
スクリプト情報利用装置３０は、スクリプト検出装置２０が検出したスクリプト情報を受信する。なお、スクリプト情報利用装置３０は、スクリプト検出装置２０で検出されたインタプリタとスクリプト情報に加えて、スクリプト検出装置２０が監視エージェント１１から取得したイベント情報（時刻情報やプロセスＩＤ（Identification）等）を、スクリプト検出装置２０から受信するようにしてもよい。
【００３３】
スクリプト情報利用装置３０は、スクリプト検出装置２０から受信したスクリプト情報等を活用して動作ログ（例えばプロセスの振る舞いの履歴等）の記録や分析、異常検知等の処理を行う。なお、スクリプト情報利用装置３０には、スクリプト情報の利用形態（例えば監視対象システム１０の状態監視、ソフトウェア障害検出、不正アクセスや情報漏洩等のインシデント検知、システム最適化ための分析等）に応じたデータ管理ツールや解析ツール等を実装するようにしてもよい。
【００３４】
監視対象システム１０は、例えばメモリに格納されたプログラムを実行するプロセッサを備え、通信ネットワークを介してスクリプト検出装置２０に通信接続される端末装置（ＰＣ（Personal Computer）、又は携帯端末等の情報通信装置等）を含む。
【００３５】
監視対象システム１０の監視エージェント１１は、監視対象システム１０のプロセスの振る舞いを監視して収集し、スクリプト検出装置２０へ送信する。監視エージェント１１は、監視対象システム１０（端末装置等）に実装され、監視対象システム１０上のプロセスを監視しリアルタイムで時系列順に当該プロセスのイベント情報を生成する構成としてもよい。特に制限されないが、監視エージェント１１は、例えばプロセスによるファイルシステムへのアクセスやネットワークへのアクセス、及び、プロセスの状態（親プロセス、子プロセスの生成・破棄等）をモニタリングし、状態が変化した場合、そのイベントを収集してスクリプト検出装置２０に送信する。監視エージェント１１で監視するプロセスは仮想マシン上のＯＳ（Operating System）で稼働するプロセスであってもよい。
【００３６】
以下、スクリプト検出装置２０についてさらに詳細に説明する。
【００３７】
図１を参照すると、スクリプト検出装置２０は、モデル作成部２１と、モデル記憶部２２と、モデル分析部２３と、分析結果記憶部２４と、スクリプト検出部２５とを備えている。モデル記憶部２２と分析結果記憶部２４は、それぞれデータベース（DataBase:DB）として構成してもよい。この場合、モデル記憶部２２、分析結果記憶部２４は、モデルＤＢ、分析結果ＤＢとも称呼してもよい。
【００３８】
特に制限されないが、以下の例では、監視するプロセスの振る舞いとして、
・プロセスによる子プロセスの実行、及び、
・プロセスによるファイルの読み込み（ｒｅａｄ）、書き込み（ｗｒｉｔｅ）を、イベントとして説明する。プロセス間通信やネットワークの通信等の監視可能なプロセスの振る舞いを分析対象に含めるようにしてもよい。この場合、検出精度をさらに高めることができる。
【００３９】
モデル作成部２１は、監視エージェント１１から受信したイベントからプロセスの振る舞いを組み立てる。そして、モデル作成部２１は、分析のための振る舞いモデルを作成し

てモデル記憶部２２へ登録する。
【００４０】
モデル分析部２３は、モデル記憶部２２に蓄積された振る舞いモデルを非同期で分析し、インタプリタとスクリプトを特定し、その振る舞い情報と合わせて分析結果記憶部２４へ登録する。
【００４１】
スクリプト検出部２５は、監視エージェント１１から取得したイベント情報を分析結果記憶部２４の情報と照らし合わせて分析する。スクリプト検出部２５は、分析の結果、スクリプトが実行されたことを検出すると、スクリプト情報利用装置３０へ通知する。
【００４２】
監視エージェント１１は、監視対象システム１０で観測したプロセスの振る舞いを監視し、監視結果をイベント情報として収集する。特に制限されないが、該イベント情報には、例えば、
・イベントを発生させたプロセスの情報、
・操作、
・操作対象、及び、
・イベントの発生順序
の情報が含まれる。
【００４３】
プロセスの情報には、特に制限されないが、例えば、
・プロセスの実態となるプログラムファイルのパス、及び、
・プロセスを一意に識別可能なＩＤ（Identification：識別情報）
が含まれる。なお、プロセスは、プロセッサ上でプログラムを動作させる際にプロセッサが実行するひとまとまりの処理単位のことを意味してもよい（プログラムは複数のプロセスを稼働可能）。
【００４４】
操作としては、特に制限されないが、例えば、
・ファイルのｒｅａｄ、ｗｒｉｔｅ、
・プログラムを子プロセスとして実行するｅｘｅｃｕｔｅ
があるものとする。
【００４５】
操作対象は、特に制限されないが、例えば操作（ｒｅａｄ／ｗｒｉｔｅ、ｅｘｅｃｕｔｅ）の対象となるファイルや子プロセス（プログラム）のファイルパス名等の情報で規定される。
【００４６】
イベントの発生順序は、時刻（イベント発生時刻）で代替するようにしてもよい。
【００４７】
監視エージェント１１は、収集したイベント情報を、通信手段（不図示）を介して、スクリプト検出装置２０へ送信する。特に制限されないが、監視エージェント１１は、スクリプト検出装置２０からのポーリングに対する応答として、収集したイベント情報をスクリプト検出装置２０に送信するようにしてもよい。この場合、ポーリング間隔の間、監視エージェント１１では、プロセスの監視の結果収集したイベント情報を監視対象システム１０（端末装置等）内のメモリにログしておく構成としてもよい。なお、監視エージェント１１でのプロセスの状態監視の条件や、ポーリング間隔等は適宜設定（変更）自在としてもよい。
【００４８】
スクリプト検出装置２０において、通信ネットワーク等を介して監視エージェント１１から受信したイベント情報は、モデル作成部２１とスクリプト検出部２５とで利用される。
【００４９】
モデル作成部２１は、監視エージェント１１から受信した個々のイベントを組み立て、

モデルを作成する。
【００５０】
図２は、モデル作成部２１の処理を模式的に説明する図である。図２において、〇で囲んだＰ１、Ｐ２はプログラム（実行ファイル）、□で囲んだＦ１はデータファイルを表している。
【００５１】
プログラムＰ１から起動されたプロセスによるファイルＦ１の読み込み（ｒｅａｄ）／書き込み（ｗｒｉｔｅ）、及び、プログラムＰ２の実行（ｅｘｅｃｕｔｅ）を、図２のように表すことにする。
【００５２】
ここで、モデル作成部２１で一連のイベントを組み立てた結果、例えば図３のようなプロセスの振る舞いになったとする。
【００５３】
図３では、
・プログラムＰ１が、プログラムＰ２を実行する（ｅｘｅｃｕｔｅ）。
・プログラムＰ２は、ファイルＦ１への読み込み／書き込み（ｒｅａｄ／ｗｒｉｔｅ）、ファイルＦ２の読み込み（ｒｅａｄ）、その後、プログラムＰ３を実行する（ｅｘｅｃｕｔｅ）。
・プログラムＰ３は、ファイルＦ３の読み込み（ｒｅａｄ）とファイルＦ４の書き込み（ｗｒｉｔｅ）を行う。
【００５４】
モデル作成部２１は、図３に示した振る舞いから分析用のモデルを作成する。具体的には、子プロセスの振る舞いは、親プロセスの振る舞いであるとみなし、図４に例示するような、振る舞いモデルを作成する。
【００５５】
図４の例では、モデル作成部２１は、プログラムＰ３のプロセスの振る舞いは親プロセスの振る舞いでもあるとみなして、プログラムＰ２のプロセスの振る舞いモデルを作成している。
【００５６】
同様に、モデル作成部２１は、プログラムＰ２の振る舞いは、プログラムＰ１の振る舞いでもあるとみなして、プログラムＰ１のモデルを作成している。
【００５７】
モデル作成部２１では、プログラムＰ１のモデルに、
・Ｐ１の振る舞い（プログラムＰ２の実行（ｅｘｅｃｕｔｅ））のほか、
・Ｐ２の振る舞い（ファイルＦ１への読み込み／書き込み（ｒｅａｄ／ｗｒｉｔｅ）、ファイルＦ２の読み込み（ｒｅａｄ）、その後、プログラムＰ３を実行（ｅｘｅｃｕｔｅ））、
・Ｐ３の振る舞い（ファイルＦ３の読み込み（ｒｅａｄ）とファイルＦ４の書き込み（ｗｒｉｔｅ））を展開して埋め込んでいる。
【００５８】
モデル作成部２１において、このように加工を行うのは、子プロセスを多用し、目的の処理を子プロセスに実行させる特徴を持つスクリプトがあるためである。
【００５９】
次に、モデル作成部２１は、作成したモデルをモデル記憶部２２へ登録する。
【００６０】
図４に例示したモデルに関して、モデル作成部２１で作成したモデルをモデル記憶部２２へ登録する例として、例えば図５に例示するようなテーブル形式で登録してもよい。図５において、モデル主体ＩＤと実主体ＩＤの列（column あるいは欄（field））は、同じプログラムが複数回起動されたときのプロセスを識別するためのＩＤである。
【００６１】

図５において、モデル主体ＩＤは、各振る舞いを行ったとみなすプロセスを示しており、図４に対応する。実主体ＩＤは、各振る舞いを実際に行ったプロセスを示しており、図３に対応する。主体プログラムは該プロセスを生成したプログラムである。発生順序、操作、操作対象は、前述したイベントの発生順序、操作、操作対象に対応する。例えば図５の一行目は、図４のモデルにおいて、プログラムＰ１がモデル主体ＩＤ、実主体ＩＤであり、主体プログラムであるＰ１が発生順序：１番目で、操作対象：Ｐ２に対する操作：ｅｘｅｃｕｔｅを行うことを表している。

【００６２】

図３と図６のプロセスの振る舞いが監視エージェント１１によって複数回観測され、監視エージェント１１からイベント情報を受信したモデル作成部２１がモデル化してモデル記憶部２２へ登録したとする。図６に示したプロセスの振る舞いは、モデル作成部２１において、前述した処理を行うことで、図７のようなモデルとしてモデル化される。

【００６３】

モデル記憶部２２へモデルが蓄積されると、モデル分析部２３は分析を開始する。

【００６４】

図８は、モデル分析部２３の処理手順を説明する流れ図である。モデル分析部２３は、モデル記憶部２２にモデルが蓄積されたプログラムを探し、分析対象のプログラムを選択する（Ｓ１０１）。ここで、プログラムＰ２が、分析対象プログラムとして選択されたとする。

【００６５】

次に、モデル分析部２３は、分析対象プログラムのモデルをモデル記憶部２２から取得し、共通の振る舞いを検出する（Ｓ１０２）。

【００６６】

一例として、モデル記憶部２２へ登録された図４と図７のモデルにおいて、プログラムＰ２のモデルを比較すると、

・ファイルＦ１に対する操作：ｒｅａｄ／ｗｒｉｔｅ、
・プログラムＰ３に対する操作：ｅｘｅｃｕｔｅ、及び、
・ファイルＦ３に対する操作：ｒｅａｄ

が共通している。

【００６７】

プログラムＰ２がインタプリタであると仮定した場合、これらの共通の振る舞いは、スクリプトの違いによらず、常に行われる振る舞いであると考えられる。一般的なプログラムに例えると、ライブラリのロードや設定の読み込み、ログの出力などが挙げられる。以降、このような振る舞いを、本明細書では、「インタプリタの共通の振る舞い」と呼ぶ。

【００６８】

次に、モデル分析部２３は、分析対象プログラムのモデル群をモデルの類似度でグループ化する（Ｓ１０３）。

【００６９】

モデルのグループ化は、インタプリタが同じスクリプトを実行しているならば、その振る舞いは類似する（類似するモデルになる）はずであるという前提に基づいて行われる。ただし、前の手順（Ｓ１０２）で検出したインタプリタの共通の振る舞いは、スクリプトの違いに関わらず類似するため、グループ化の判定基準から除外する。

【００７０】

モデル分析部２３におけるモデルのグループ化の一例として、例えば、モデルの類似度を距離として数値化することによって既存のクラスタリング手法（教師なし機械学習：データの集まりをデータ間の類似度等に従っていつかのグループに分ける手法）を適用することができる。

【００７１】

類似度の距離は、例えば、

モデルの振る舞いの操作対象が同じならばモデル間の距離は近い、

モデルの振る舞いの操作対象に対する操作（ｒｅａｄ／ｗｒｉｔｅ／ｅｘｅｃｕｔｅ）が同じならばモデル間の距離は近い、
モデルの振る舞いの順序が同じならばモデル間の距離は近い、
などの基準を用いてもよい。あるいは、監視対象システム１０に合わせて、これらの基準を適宜調整（チューニング）するようにしてもよい。

【００７２】

モデル分析部２３では、モデルの類似度に基づくグループ化の結果、類似するモデルが無い場合には（Ｓ１０４の分岐Ｂ）、分析対象プログラムがインタプリタではないと判断し、当該分析対象プログラムの分析を終了する。

【００７３】

類似するモデルが無いことは、
・類似度の低いモデルが同一グループになる、又は
・モデル数の少ないグループに分かれる、
ことから判断することができる。

【００７４】

モデル分析部２３において、類似するモデルでグループ化できた場合（Ｓ１０４の分岐Ａ）、分析対象プログラムがインタプリタであり、それぞれのグループがスクリプトを実行していると判断して、スクリプトの特定を行う（Ｓ１０５）。

【００７５】

図４のプログラムＰ２のモデルを分析してスクリプトの特定を行う手順（図８のＳ１０５）を以下に説明する。

【００７６】

図４に示したプログラムＰ２のモデルの振る舞いのうち、
・ファイルＦ１に対する操作：ｒｅａｄ／ｗｒｉｔｅ、
・プログラムＰ３に対する操作：ｅｘｅｃｕｔｅ、
・ファイルＦ３に対する操作：ｒｅａｄ、
は、インタプリタの共通の振る舞いであるとわかっている。このため、図４において、これらを分析の対象から外すと、図９の太線で示した振る舞いが残る。

【００７７】

インタプリタはスクリプトを実行する前に当該スクリプトを読み込むはずであるという前提に基づき、図９の残された振る舞い（太線）のうち、プログラムＰ２が最初に読み込みを行ったファイルＦ２を、スクリプトとして特定することができる。

【００７８】

つまり、プログラムＰ２は、ファイルＦ２の入力によって、その後の振る舞いが変化させられたと見做すことができる。

【００７９】

同様に、図７のプログラムＰ２のモデルから、インタプリタの共通の振る舞いを除外すると、図１０の太線の振る舞いが残る。図１０において、太線で示す残りの振る舞いから、プログラムＰ２によって最初に読み込まれたファイルＦ５がスクリプトであると特定することができる。

【００８０】

次に、モデル分析部２３において、上記と同様の分析を、他のプログラムＰ３とＰ１に対しても適用する場合について説明する。図４及び図７のＰ３のモデルを分析した場合、グループ化の結果によっては、プログラムＰ３がインタプリタであり、ファイルＦ３かＦ６がスクリプトであると、モデル分析部２３によって特定される可能性がある。

【００８１】

しかし、インタプリタが同じスクリプトを実行しているならば、その振る舞いは類似するはずである、ということを前提としたとする。この前提に基づくと、スクリプトファイルＦ２を実行しているプログラムＰ２のプロセスの子プロセスとして実行されたプログラムＰ３の振る舞いが類似するモデルになるのは必然であると考えられる。

【００８２】
このため、モデル分析部２３において、プログラムＰ２がスクリプトを実行していると判定した時点で、プログラムＰ３はスクリプトを実行していないと判定する。
【００８３】
モデル分析部２３では、スクリプトファイルＦ５の子プロセスについても、同様に判定する。
【００８４】
また、スクリプト全体の振る舞いを決めるのは、子プロセスより親側のプロセスのスクリプトであると考えられる。
【００８５】
このことから、プログラムＰ３がスクリプトを実行していたとしても、分析結果として出力する情報としては、親プロセスであるプログラムＰ２が実行しているスクリプト情報のほうが有用であると考えられる。
【００８６】
ただし、スクリプトの検出結果を利用するアプリケーション（スクリプト情報利用装置３０上のアプリケーション）によっては、プログラムＰ３が実行するスクリプトの情報が有用である可能性もあるため、プログラムＰ３がスクリプトを実行していると判定してもよい。
【００８７】
モデル分析部２３において、図４及び図７に示したプログラムＰ１のモデルを分析した場合、グループ化の結果によっては、子プロセスが読み込んだファイルのいずれかがスクリプトとして特定される可能性がある。このとき、モデル分析部２３において、プログラムＰ１のモデルの振る舞いから、プログラムＰ１が常に実行する振る舞いを除外し、残った振る舞いが図１１の太線の通りであったとする。
【００８８】
モデル分析部２３において、図１１に太線で示す残った振る舞いを実際に実行していたプログラムを確認するため、モデル記憶部２２に格納された図５の情報と照合して各振る舞いの実主体ＩＤを確認する。その結果、図１１に太線で示す残った振る舞いを実際に実行したプロセスは、図１２に太線で示す振る舞いのように、プログラムＰ２の配下に閉じていることがわかる。
【００８９】
このことから、モデル分析部２３では、プログラムＰ１はスクリプトを実行していないと判断する。
【００９０】
プログラムＰ１がスクリプトを実行していたとしても、スクリプト全体の振る舞いに影響を与えない、と考えられる。このため、モデル分析部２３が分析結果として出力する情報としては、事実上の振る舞いを決定しているプログラムＰ２のスクリプト情報の方が有用である、と考えられる。
【００９１】
ただし、スクリプトの検出結果を利用するアプリケーション（スクリプト情報利用装置３０上のアプリケーション）によっては、プログラムＰ１が実行するスクリプトの情報が有用である可能性もある。このため、例えばスクリプト検出結果の利用等に応じて、モデル分析部２３では、プログラムＰ１がスクリプトを実行していると判定してもよい。
【００９２】
モデル分析部２３は、特定されたスクリプトと周辺プロセスとの関係として、該スクリプトを実行する親プロセスが該スクリプトの振る舞いに影響を与えていない場合（Ｓ１０６の分岐Ｄ）、分析を終了する。
【００９３】
次に、モデル分析部２３は、特定したスクリプトのモデルのグループを分析し、グループ内で共通の振る舞いを抽出する（Ｓ１０７）。プログラムが同じスクリプトを実行して

いるならば、その振る舞いは類似するモデルになるという前提があるが、全く同じになるわけではない。このため、モデル分析部２３では、特定したスクリプトが実行されるごとに異なっていた振る舞いを除外する。その結果、残された振る舞いは、特定したスクリプトがグループ内で常に実行する振る舞いであると考えられる。スクリプトがグループ内で常に実行する振る舞いを「スクリプトの共通の振る舞い」と呼ぶことにする。

【００９４】

１つのスクリプトの振る舞いが大幅に変わる場合には、当該１つのスクリプトに対して複数の振る舞いのグループに分けられる場合もあるが、本手順の通りに、分析可能である。

【００９５】

次に、モデル分析部２３は、分析結果を分析結果記憶部２４へ格納する（Ｓ１０８）。具体的には、モデル分析部２３は、

・インタプリタの共通の振る舞い、及び、

・インタプリタとスクリプトの組み合わせにおけるスクリプトの共通の振る舞い、

を格納する。

【００９６】

図１３（Ａ）、（Ｂ）は、分析結果記憶部２４に記憶された分析結果の一例を説明する図である。図１３（Ａ）において、テーブル４０１は、インタプリタの共通の振る舞いを記憶する。図１３（Ｂ）において、テーブル４０２は、インタプリタとスクリプトの組み合わせにおけるスクリプトの共通の振る舞いを記憶する。

【００９７】

図１３（Ａ）の例では、テーブル４０１において、インタプリタ（プログラム）Ｐ２は、発生順序：１で操作対象：ファイルＦ１に操作：ｒｅａｄ／ｗｒｉｔｅを行い、発生順序：２で操作対象：プログラムＰ３に操作：ｅｘｅｃｕｔｅを行い、発生順序：３で操作対象：ファイルＦ３に操作：ｒｅａｄを行っている。

【００９８】

また、図１３（Ｂ）の例では、テーブル４０２において、インタプリタ（プログラム）Ｐ２は、スクリプト（スクリプトファイル）Ｆ２を読み込んで実行し、操作対象：ファイルＦ４に操作：ｗｒｉｔｅを発生順序：１で行っている。また、インタプリタＰ２は、スクリプト（スクリプトファイル）Ｆ５を読み込んで実行し、操作対象：ファイルＦ６に操作：ｒｅａｄ／ｗｒｉｔｅを発生順序：１で行っている。なお、ファイルＦ２、Ｆ５等、インタプリタで読み込まれるスクリプトファイルは、単にスクリプトともいう。

【００９９】

この時点（モデル分析部２３が分析結果を分析結果記憶部２４へ格納した時点）で、過去に実行されたインタプリタＰ２とスクリプトＦ２、Ｆ５が特定されている。このため、スクリプト情報利用装置３０が許容するのであれば、スクリプト検出装置２０は、モデル分析部２３で特定されたスクリプトの情報をスクリプト情報利用装置３０に出力しても良い。

【０１００】

ただし、インタプリタとスクリプトの特定を行うために、スクリプトが複数回実行されるだけの時間が経過しているため、古い情報となっている。

【０１０１】

次に、本実施形態において、モデル分析部２３によって特定されたスクリプトの情報が分析結果記憶部２４に格納された後の動作を説明する。

【０１０２】

監視対象システム１０において、再びプログラムＰ１が実行され、図３の通りに動作すると、監視エージェント１１からスクリプト検出装置２０へイベント情報が送信される。スクリプト検出部２５とモデル作成部２１が該イベント情報を受信する。モデル作成部２１は、該イベント情報を受信すると、前述の通り動作する。

【０１０３】

スクリプト検出部２５は、図１４の処理手順に従って、スクリプトが実行されたことを

検出する。
【０１０４】
図１４を参照すると。最初に、スクリプト検出部２５は、実行されたプログラムＰ１、Ｐ２、Ｐ３の情報が、分析結果記憶部２４に登録されていることを確認する（Ｓ２０１）。
【０１０５】
ここで、分析結果記憶部２４には、図１３（Ａ）のテーブル４０１のように、プログラムＰ２のインタプリタの共通の振る舞いが登録されているが、プログラムＰ１とＰ３の情報は登録されていなかったとする。
【０１０６】
スクリプト検出部２５は、分析結果記憶部２４にインタプリタの共通の振る舞いが登録されている場合（Ｓ２０２の分岐Ａ）、当該プログラム（この場合、プログラムＰ２）が実行するスクリプトを検出することができると判断し、分析を行う。
【０１０７】
一方、スクリプト検出部２５は、分析結果記憶部２４にインタプリタの共通の振る舞いが登録されていない場合、当該プログラム（この場合、プログラムＰ１とＰ３）の分析は行わない（Ｓ２０２の分岐Ｂ）。
【０１０８】
スクリプト検出部２５は、モデル作成部２１と同様の手順で振る舞いのモデルを組み立て、例えば図４のプログラムＰ２のモデルを作成する（Ｓ２０３）。
【０１０９】
次に、スクリプト検出部２５は、図１３（Ａ）の分析結果記憶部２４のテーブル４０１を参照して、プログラムＰ２のインタプリタの共通の振る舞いをモデルから除外し（Ｓ２０４）、図９の実線で示すモデルを得る。図４のプログラムＰ２のモデルから、プログラムＰ２のインタプリタの共通の振る舞い（順序１でファイルＦ１に対してｒｅａｄ／ｗｒｉｔｅ、順序２でプログラムＰ３をｅｘｅｃｕｔｅ、順序３でファイルＦ３に対してｒｅａｄ）を削除すると、図９に実線で示すプログラムＰ２のモデル（順序１でファイルＦ２に対してｒｅａｄ、順序２でファイルＦ４に対してｗｒｉｔｅ）を得る。
【０１１０】
スクリプト検出部２５は、図９のモデルに基づき、前述したモデル分析部２３と同様の方法で、残された振る舞いの中で、最初に読み込まれたファイルＦ２がスクリプトである可能性が高いと判断する（Ｓ２０５）。
【０１１１】
この時点で、スクリプト検出部２５は、インタプリタＰ２によりスクリプトＦ２が実行されたことを、スクリプト情報利用装置３０へ確度の低い情報として通知してもよい。
【０１１２】
この方法によれば、スクリプト検出部２５がモデルを完成させる前であっても、インタプリタＰ２がスクリプトＦ２を読み込んだことが判明した時点で通知できる。このため、ヒューリスティックであるが、スクリプト検出部２５は、スクリプト情報利用装置３０に対して、リアルタイムに近い通知が可能である（Ｓ２０６の分岐Ｄ）。
【０１１３】
次に、スクリプト検出部２５は、図１３（Ｂ）の分析結果記憶部２４のテーブル４０２を参照して、インタプリタＰ２とスクリプトＦ２の組み合わせにおけるスクリプトの共通の振る舞いを取得する（Ｓ２０７）。
【０１１４】
ここで、分析中のプログラムが分析結果記憶部２４にはインタプリタとして登録されているにもかかわらず、特定したスクリプトが分析結果記憶部２４に登録されていない場合に、初めて実行されたスクリプトであると判定し、スクリプト情報利用装置３０へ通知する。
【０１１５】
例えば、分析中のプログラムがＰ２であり、スクリプト検出部２５が特定したスクリプ

トがファイルＦ３であったとする。
【０１１６】
図１３（Ａ）の分析結果記憶部２４のテーブル４０１には、プログラムＰ２がインタプリタとして登録されているが、図１３（Ｂ）の分析結果記憶部２４のテーブル４０２には、ファイルＦ３がスクリプトとして登録されていない。このため、スクリプト検出部２５では、ファイルＦ３は初めて実行されたスクリプトであると判定する。
【０１１７】
スクリプト検出部２５におけるこの判定方法は、インタプリタが既知のスクリプトを読み込んだ時点でスクリプトが実行されたと判断する方法と同様に、ヒューリスティックな方法である。しかしながら、この判定方法によれば、監視対象システム１０においてスクリプトが複数回実行され、その振る舞いに対するモデル分析部２３による分析結果を待たずに、スクリプト検出部２５では新しく実行されたスクリプトを検出することができる。
【０１１８】
スクリプト検出部２５は、インタプリタ（プログラム）Ｐ２とスクリプトＦ２の組み合わせにおけるスクリプトの共通の振る舞いを、スクリプト検出部２５自身が組み立てたモデルと比較する（Ｓ２０８）。スクリプト検出部２５が組み立てたモデルが、図１３（Ｂ）の分析結果記憶部２４のテーブル４０２から取得したスクリプトの共通の振る舞いを包含している場合には、スクリプト検出部２５は、インタプリタ（プログラム）Ｐ２によりスクリプトファイルＦ２が実行されたと判断し（Ｓ２０８の分岐Ｅ）、スクリプト情報利用装置３０へ通知する。
【０１１９】
スクリプトの振る舞いは常に同じとは限らない。このため、分析結果記憶部２４に格納されたスクリプトの共通の振る舞い以外の振る舞いがあったとしても、スクリプト検出部２５では、これを無視する。
【０１２０】
一方で、スクリプト検出部２５が組み立てたモデルが、図１３（Ｂ）の分析結果記憶部２４のテーブル４０２から取得したスクリプトの共通の振る舞いを包含しない場合には（Ｓ２０９の分岐Ｈ）、スクリプト検出部２５では、分析対象のプログラムＰ２は既知のスクリプトを実行していないと判断する。この場合、監視エージェント１１からのイベント情報を受け取ったモデル作成部２１で新しくモデルが生成され、モデル分析部２３経由で新しく振る舞いが分析され、分析結果記憶部２４の内容が更新されるまで（当該スクリプトの共通の振る舞いが図１３（Ｂ）の分析結果記憶部２４のテーブル４０２に登録されるまで）、スクリプト検出部２５では、当該スクリプトを既知のスクリプトとしては検出しない。
【０１２１】
スクリプト検出部２５がスクリプトの実行を検出した場合に（Ｓ２０６の分岐Ｄ、Ｓ２０９の分岐Ｇ）、スクリプト情報をスクリプト情報利用装置３０へ出力する（Ｓ２１０）。
【０１２２】
スクリプト検出部２５がスクリプト情報利用装置３０に通知するスクリプト情報の内容としては、インタプリタとスクリプトのファイル名としてもよい。あるいは、監視エージェント１１からプロセスＩＤや実行時刻等の関連する情報を取得しておき、スクリプト情報利用装置３０へ通知する情報に含めるようにしてもよい。
【０１２３】
本実施形態によれば、実環境におけるプロセスの振る舞いを分析することで、プログラムの振る舞いを決める大元の入力を特定するものであり、結果として、インタプリタとスクリプトの特定を可能としている。
【０１２４】
本実施形態によれば、監視対象システム１０で実行されるプログラムのうちどのプログラムがインタプリタであり、どのファイルをスクリプトとして実行しているかを機械的に特定することを可能としている。このため、システムの動作の分析や動作ログの出力など

に実行されたスクリプトの情報を利用することができる。

【０１２５】

本実施形態において、モデルをグループ化するためのモデル間の距離の算出のために監視エージェント１１から追加の情報をスクリプト検出装置２０に送信してもよい。例えば、ユーザ間でスクリプトが共有されないシステムでは、監視エージェント１１からプログラムを実行したユーザの情報をスクリプト検出装置２０に送信し、モデル分析部２３やスクリプト検出部２５では、該プログラムを実行したユーザの違いをモデル間の距離に反映するようにしてもよい。このように、モデル間の距離の算出のための追加情報を利用することで、グループ化の精度を向上できる可能性がある。

【０１２６】

図１３（Ｂ）の分析結果記憶部２４では、テーブル４０２において、インタプリタとスクリプトの組み合わせをキーにしているが、スクリプトだけをキーとしても良い。例えば複数のバージョンのインタプリタがインストールされたシステムで、あるスクリプトを異なるバージョンのインタプリタで実行しても同じ振る舞いになると見做すことができる場合、インタプリタの違いは無視することができる。

【０１２７】

本実施形態では、スクリプトはファイルであることを前提に説明したが、実環境では、端末間通信などのファイル以外の形で得たデータがメモリ上でスクリプトとして実行される場合がある。

【０１２８】

例えば、監視エージェント１１において、端末間通信などのプロセスがアクセスしたデータを監視し、監視結果であるイベント情報をスクリプト検出装置２０に送信し、スクリプト検出装置２０でモデル化して分析することで、ファイル以外の形で存在するスクリプトを検出することが可能である。

【０１２９】

図１５は、図１のスクリプト検出装置２０をコンピュータ装置１００で構成した例を説明する図である。コンピュータ装置１００は、プロセッサ１０１、メモリ１０２、表示装置１０３、通信インタフェース１０４を備えている。メモリ１０２は、例えばＲＡＭ（Random Access Memory）、ＲＯＭ（Read Only Memory）、又は、ＥＥＰＲＯＭ（Electrically Erasable and Programmable ROM）等の半導体メモリであってもよいし、あるいは、ＨＤＤ（Hard Disk Drive）、ＳＳＤ（Solid State Drive,）、ＵＳＢ（Universal Serial Bus)、ＣＤ（Compact Disc）、ＤＶＤ（Digital Versatile Disc）等のストレージデバイスであってもよい。メモリ１０２は、プロセッサ１０１で実行させるプログラム（命令群（instructions）、データ）１１０を記憶している。通信インタフェース１０４は、ＮＩＣ（Network Interface Card）等を備え、監視対象システム１０及びスクリプト情報利用装置３０に通信ネットワークを介して通信接続する。プロセッサ１０１は複数のプロセッサ構成（マルチプロセッサ等）であってもよい。特に制限されないが、コンピュータ装置１００は、例えばサーバ装置であってもよい。この場合、コンピュータ装置１００は、例えば監視対象システム１０でのスクリプト検出による各種クラウドサービス（例えばインシデント検知サービス等）を提供するクラウドサーバ装置であってもよい。あるいは、別の形態として、コンピュータ装置１００は、監視対象システム１０に直結する構成としてもよい。

【０１３０】

例えばプロセッサ１０１がメモリ１０２から読み出して実行するプログラム１１０は、図１６に例示するように、モデル作成モジュール１１１とモデル分析モジュール１１２を含む。

【０１３１】

モデル作成モジュール１１１は、プロセスの振る舞いの監視の結果収集されたイベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成する。

【０１３２】

モデル分析モジュール１１２は、分析対象のプログラムに関するモデルについて類似度に基づきグループ化できる場合、前記分析対象のプログラムはスクリプトを読み込んで実行するインタプリタであると判断し、モデルを分析してスクリプトの特定を行う。

【０１３３】

さらに、モデル分析モジュール１１２の処理につづいて、特定されたスクリプトが分析結果記憶部（図１の２４）に格納されているスクリプトであるか否かを判別し、前記特定されたスクリプトが既知のプロセスであるか否かを判別するプログラムモジュールを含むようにしてもよい。

【０１３４】

上記各モジュールは、スクリプト検出部２５の処理に対応しているが、メモリ１０２に保持されるプログラム１１０は、スクリプト検出部２５の処理に制限されるものでないことは勿論である。例えば、プログラム１１０は、前記実施形態で説明したモデル作成部２１、及びモデル分析部２３の処理をプロセッサ１０１で実行させるためのプログラムモジュールを含むようにしてもよい。

【０１３５】

さらに、スクリプト情報利用装置３０も、図１５に例示したコンピュータ装置１００で実装するようにしてもよいことは勿論である。なお、スクリプト検出装置２０とスクリプト情報利用装置３０をバス接続、又は直接接続（例えばプロセッサ装置の直接接続）等で接続し一つの装置内に実装するようにしてもよいことは勿論である。この場合、スクリプト検出装置２０とスクリプト情報利用装置３０の処理を、１つのプロセッサで実行するか、各装置の処理を１つ又は複数のプロセッサで実行するようにしてもよい。

【０１３６】

図１において、監視対象システム１０は、複数の端末装置（情報通信装置）を備えた構成としてもよい。スクリプト検出装置２０では、端末装置毎にインストールされた監視エージェント１１からイベント情報を受信してモデルを作成し、モデルの分析結果（特定されたスクリプト情報等）を、端末装置のＩＤ（Identification）やアドレス（例えばＩＰ（Internet Protocol）アドレス）等に関連付けて（紐付けて）分析結果記憶部２４に格納するようにしてもよい。この場合、スクリプト検出部２５では、端末装置毎に、モデルを組み立てて検出したスクリプトが、分析結果記憶部２４に登録されている既知のスクリプトであるか、新しいスクリプトであるかを判別するようにしてもよい。なお、監視対象システム１０上のプロセスの振る舞いを監視するための監視エージェント１１がインストール可能であり、スクリプトを実行するものであれば、監視対象システム１０は、端末装置に制限されるものでなく、例えばルータ等のネットワーク機器や、サーバ装置等、ＩｏＴ（Internet of Things）デバイス任意の情報通信装置を含む構成としてもよい。

【０１３７】

本実施形態は、プロセスの振る舞いをモデル化して分析することでスクリプトを特定可能としているが、例えばセキュリティ分野に適用した場合に、マルウェアの検出やフォレンジック等に利用することもできる（ただし、左記に制限されない）。その他の適用分野として、システムの振る舞いの分析や最適化など、システムの運用管理などにも利用できる。

【０１３８】

なお、上記の特許文献１、２、及び非特許文献１の各開示を、本書に引用をもって繰り込むものとする。本発明の全開示（請求の範囲を含む）の枠内において、さらにその基本的技術思想に基づいて、実施形態ないし実施例の変更・調整が可能である。また、本発明の請求の範囲の枠内において種々の開示要素（各請求項の各要素、各実施例の各要素、各図面の各要素等を含む）の多様な組み合わせ乃至選択が可能である。すなわち、本発明は、請求の範囲を含む全開示、技術的思想にしたがって当業者であればなし得るであろう各種変形、修正を含むことは勿論である。

【０１３９】

上記した実施形態は、例えば以下のように付記される（ただし、以下に制限されない）。

【０１４０】
（付記１）
監視対象システム上のプロセスの振る舞いの監視の結果収集されたイベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成する第１の手段と、
分析対象のプログラムに関するモデルについて類似度に基づきグループ化できる場合、前記分析対象のプログラムはスクリプトを読み込んで実行するインタプリタであると判断しスクリプトの特定を行う第２の手段と、
を含む、スクリプト検出装置。
【０１４１】
（付記２）
前記第２の手段で特定されたスクリプトが記憶部に格納されているスクリプトであるか否かを判別し、前記特定されたスクリプトが既知のプロセスであるか否かを判別する第３の手段をさらに含む、付記１に記載のスクリプト検出装置。
【０１４２】
（付記３）
前記第１乃至第３の手段を備えたスクリプト検出部を備え、さらに、
前記第１の手段と同様に、前記イベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成するモデル作成部と、
前記第２の手段と同様にして、スクリプトを特定し、特定したスクリプトの情報を前記記憶部に格納するモデル分析部と、
前記記憶部と、
を含む、付記２に記載のスクリプト検出装置。
【０１４３】
（付記４）
前記第２の手段は、類似したモデルでグループ化できた場合、前記インタプリタの共通の振る舞いとして、前記インタプリタによる操作と、操作対象を関連付けて前記記憶部に記憶し、
さらに、前記スクリプトの共通の振る舞いとして、前記スクリプトと、前記スクリプトを実行するインタプリタによる操作、操作対象を関連付けて前記記憶部に記憶する、付記２又は３に記載のスクリプト検出装置。
【０１４４】
（付記５）
前記第３の手段は、前記第１の手段で組み立てた前記モデルに関して、前記スクリプトの振る舞いが、前記記憶部から取得した前記スクリプトの共通の振る舞いを包含している場合には、前記インタプリタである前記分析対象のプログラムにより前記特定したスクリプトファイルが実行されたと判断し、前記スクリプトの振る舞いが、前記記憶部から取得した前記スクリプトの共通の振る舞いを包含しない場合には、前記分析対象のプログラムは、既知のスクリプトを実行していないと判断する、付記４に記載のスクリプト検出装置。
【０１４５】
（付記６）
前記第２の手段は、類似度に基づきグループ化を行うにあたり、スクリプトの相違によらずインタプリタで共通に行われる振る舞いを外した上で前記グループ化を行う、付記１乃至５のいずれかに記載のスクリプト検出装置。
【０１４６】
（付記７）
前記第２の手段は、子プロセスの振る舞いを、親プロセスである大元のプログラムの振る舞いに埋め込み、前記大元のプログラムの時間軸上の振る舞いとして展開したモデルを生成する、付記１乃至６のいずれかに記載のスクリプト検出装置。
【０１４７】
（付記８）

前記第２の手段は、子プロセスであるプログラムが第１のスクリプトを実行していると判断し、さらに、前記子プロセスの親プロセスであるプログラムが第２のスクリプトを実行していると判断した場合、スクリプト情報の有用性に基づき、前記第１及び第２のスクリプトのいずれか一方のスクリプトを選択して前記記憶部に記憶する、付記１乃至７のいずれかに記載のスクリプト検出装置。

【０１４８】

（付記９）

前記第２の手段は、子プロセスであるプログラムが第１のスクリプトを実行していると判断し、さらに、前記子プロセスの親プロセスであるプログラムが第２のスクリプトを実行していると判断した場合、スクリプト全体の振る舞いに与える影響に基づき、スクリプトを選択して前記記憶部に記憶する、付記１乃至８のいずれかに記載のスクリプト検出装置。

【０１４９】

（付記１０）

付記１乃至９のいずれかに記載の前記スクリプト検出装置と、

前記スクリプト検出装置に通信接続され、前記スクリプト検出装置からスクリプト検出結果を受け取り、前記スクリプト検出結果を利用して前記監視対象システムに関する分析を行うスクリプト情報利用装置と、

を備えた管理システム。

【０１５０】

（付記１１）

プロセスの振る舞いの監視の結果収集されたイベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成する第１の処理と、

分析対象のプログラムに関するモデルについて類似度に基づきグループ化できる場合、前記分析対象のプログラムはスクリプトを読み込んで実行するインタプリタであると判断しスクリプトの特定を行う第２の処理を含む、スクリプト検出方法。

【０１５１】

（付記１２）

前記第２の処理で特定されたスクリプトが記憶部に格納されているスクリプトであるか否かを判別し、前記特定されたスクリプトが既知のプロセスであるか否かを判別する第３の処理をさらに含む、付記１１に記載のスクリプト検出方法。

【０１５２】

（付記１３）

前記第１乃至第３の処理をスクリプト検出部で実行し、

前記第１の処理と同様に、前記イベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成するモデル作成処理と、

前記第２の処理と同様にして、スクリプトを特定し、特定したスクリプトの情報を前記記憶部に格納するモデル分析処理を含む、付記１２に記載のスクリプト検出方法。

【０１５３】

（付記１４）

前記第２の処理は、類似したモデルでグループ化できた場合、前記インタプリタの共通の振る舞いとして、前記インタプリタによる操作と、操作対象を関連付けて前記記憶部に記憶し、

さらに、前記スクリプトの共通の振る舞いとして、前記スクリプトと、前記スクリプトを実行するインタプリタによる操作、操作対象を関連付けて前記記憶部に記憶する、付記１２又は１３に記載のスクリプト検出方法。

【０１５４】

（付記１５）

前記第３の処理は、前記第１の処理で組み立てた前記モデルに関して、前記スクリプトの振る舞いが、前記記憶部から取得した前記スクリプトの共通の振る舞いを包含している

場合には、前記インタプリタである前記分析対象のプログラムにより前記特定したスクリプトファイルが実行されたと判断し、前記スクリプトの振る舞いが、前記記憶部から取得した前記スクリプトの共通の振る舞いを包含しない場合には、前記分析対象のプログラムは、既知のスクリプトを実行していないと判断する、付記１４に記載のスクリプト検出方法。

【０１５５】

（付記１６）

前記第２の処理は、類似度に基づきグループ化を行うにあたり、スクリプトの相違によらずインタプリタで共通に行われる振る舞いを外した上で前記グループ化を行う、付記１１乃至１５のいずれかに記載のスクリプト検出方法。

【０１５６】

（付記１７）

前記第２の処理は、子プロセスの振る舞いを、親プロセスである大元のプログラムの振る舞いに埋め込み、前記大元のプログラムの時間軸上の振る舞いとして展開したモデルを生成する、付記１１乃至１６のいずれかに記載のスクリプト検出方法。

【０１５７】

（付記１８）

前記第２の処理は、子プロセスであるプログラムが第１のスクリプトを実行していると判断し、さらに、前記子プロセスの親プロセスであるプログラムが第２のスクリプトを実行していると判断した場合、スクリプト情報の有用性に基づき、前記第１及び第２のスクリプトのいずれか一方のスクリプトを選択して前記記憶部に記憶する、付記１１乃至１７のいずれかに記載のスクリプト検出方法。

【０１５８】

（付記１９）

前記第２の処理は、子プロセスであるプログラムが第１のスクリプトを実行していると判断し、さらに、前記子プロセスの親プロセスであるプログラムが第２のスクリプトを実行していると判断した場合、スクリプト全体の振る舞いに与える影響に基づき、スクリプトを選択して前記記憶部に記憶する、付記１１乃至１８のいずれかに記載のスクリプト検出方法。

【０１５９】

（付記２０）

プロセスの振る舞いの監視の結果収集されたイベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成する第１の処理と、

分析対象のプログラムに関するモデルについて類似度に基づきグループ化できる場合、前記分析対象のプログラムはスクリプトを読み込んで実行するインタプリタであると判断しスクリプトの特定を行う第２の処理と、

をコンピュータに実行させるプログラム。

【０１６０】

（付記２１）

前記第２の処理で特定されたスクリプトが記憶部に格納されているスクリプトであるか否かを判別し、前記特定されたスクリプトが既知のプロセスであるか否かを判別する第３の処理をさらに前記コンピュータに実行させる付記２０に記載のプログラム。

【０１６１】

（付記２２）

前記第１乃至第３の処理がスクリプト検出モジュールを構成し、

前記第１の処理と同様に、前記イベント情報を受信し、前記イベント情報に基づきプロセスの振る舞いのモデルを作成するモデル作成処理と、

前記第２の処理と同様にして、スクリプトを特定し、特定したスクリプトの情報を前記記憶部に格納するモデル分析処理をさらに含む、付記２１に記載のプログラム。

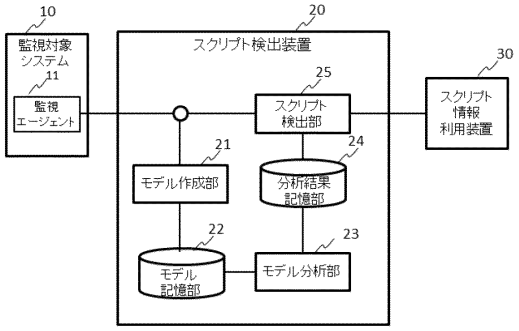
【０１６２】

（付記２３）
前記第２の処理は、類似したモデルでグループ化できた場合、前記インタプリタの共通の振る舞いとして、前記インタプリタによる操作と、操作対象を関連付けて前記記憶部に記憶し、
さらに、前記スクリプトの共通の振る舞いとして、前記スクリプトと、前記スクリプトを実行するインタプリタによる操作、操作対象を関連付けて前記記憶部に記憶する、付記２１又は２２に記載のプログラム。
【０１６３】
（付記２４）
前記第３の処理は、前記第１の処理で組み立てた前記モデルに関して、前記スクリプトの振る舞いが、前記記憶部から取得した前記スクリプトの共通の振る舞いを包含している場合には、前記インタプリタである前記分析対象のプログラムにより前記特定したスクリプトファイルが実行されたと判断し、前記スクリプトの振る舞いが、前記記憶部から取得した前記スクリプトの共通の振る舞いを包含しない場合には、前記分析対象のプログラムは、既知のスクリプトを実行していないと判断する、付記２３に記載のプログラム。
【０１６４】
（付記２５）
前記第２の処理は、類似度に基づきグループ化を行うにあたり、スクリプトの相違によらずインタプリタで共通に行われる振る舞いを外した上で前記グループ化を行う、付記２０乃至２４のいずれかに記載のプログラム。
【０１６５】
（付記２６）
前記第２の処理は、子プロセスの振る舞いを、親プロセスである大元のプログラムの振る舞いに埋め込み、前記大元のプログラムの時間軸上の振る舞いとして展開したモデルを生成する、付記２０乃至２５のいずれかに記載のプログラム。
【０１６６】
（付記２７）
前記第２の処理は、子プロセスであるプログラムが第１のスクリプトを実行していると判断し、さらに、前記子プロセスの親プロセスであるプログラムが第２のスクリプトを実行していると判断した場合、スクリプト情報の有用性に基づき、前記第１及び第２のスクリプトのいずれか一方のスクリプトを選択して前記記憶部に記憶する、付記２０乃至２６のいずれかに記載のプログラム。
【０１６７】
（付記２８）
前記第２の処理は、子プロセスであるプログラムが第１のスクリプトを実行していると判断し、さらに、前記子プロセスの親プロセスであるプログラムが第２のスクリプトを実行していると判断した場合、スクリプト全体の振る舞いに与える影響に基づき、スクリプトを選択して前記記憶部に記憶する、付記２０乃至２７のいずれかに記載のプログラム。
【０１６８】
（付記２９）
付記２０乃至２８のいずれかに記載のプログラムを記憶した非一時的なコンピュータ可読媒体。
【符号の説明】
【０１６９】
１０　監視対象システム
１１　監視エージェント
２０　スクリプト検出装置
２１　モデル作成部
２２　モデル記憶部
２３　モデル分析部

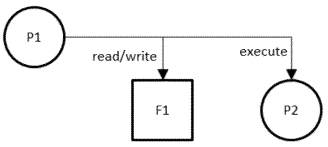
２４　分析結果記憶部
２５　スクリプト検出部
３０　スクリプト情報利用装置
１００　コンピュータ装置
１０１　プロセッサ
１０２　メモリ
１０３　表示装置
１０４　通信インタフェース
１１０　プログラム
１１１　モデル作成モジュール
１１２　モデル分析モジュール

【図面】

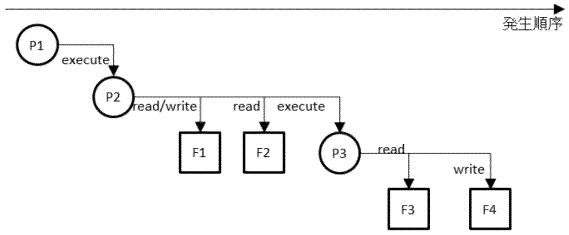
【図１】



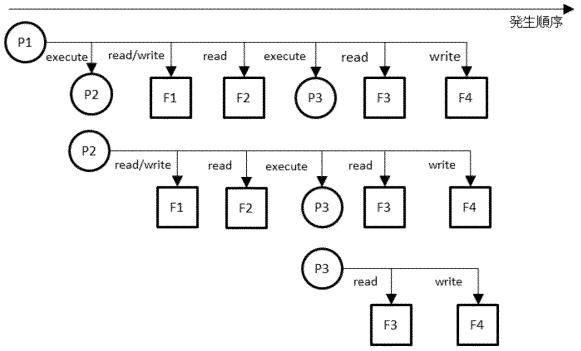
【図２】



【図３】



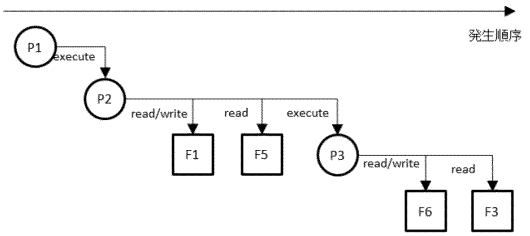
【図４】



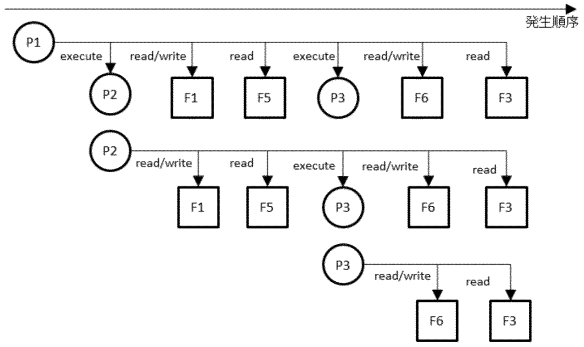
【図５】

モデル主体ID	実主体ID	主体プログラム	発生順序	操作	操作対象
100	100	P1	1	execute	P2
100	200	P1	2	read/write	F1
100	200	P1	3	read	F2
100	200	P1	4	execute	P3
100	300	P1	5	read	F3
100	300	P1	6	write	F4
200	200	P2	1	read/write	F1
200	200	P2	2	read	F2
200	**200**	P2	3	execute	P3
200	300	P2	4	read	F3
200	300	P2	5	write	F4
300	300	P3	1	read	F3
300	300	P3	2	write	F4

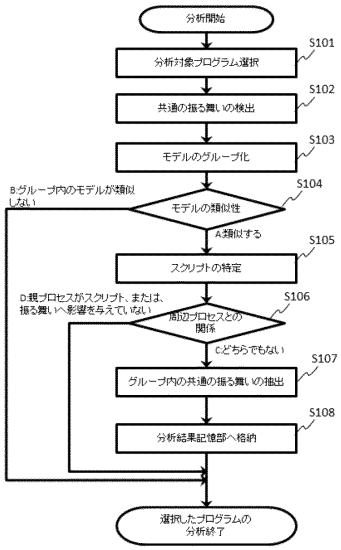
【図６】



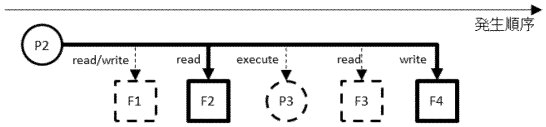
【図７】


発生順序
P1
execute
read/write
read
execute
read/write
read
P2
F1
F5
P3
F6
F3
P2
read/write
read
execute
read/write
read
F1
F5
P3
F6
F3
P3
read/write
read
F6
F3

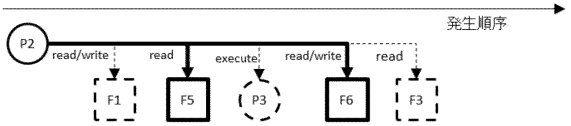
【図８】


分析開始
S101
分析対象プログラム選択
S102
共通の振る舞いの検出
S103
モデルのグループ化
S104
モデルの類似性
B.グループ内のモデルが類似しない
A.類似する
S105
スクリプトの特定
S106
D.親プロセスがスクリプト、または、振る舞いへ影響を与えていない
周辺プロセスとの関係
C.どちらでもない
S107
グループ内の共通の振る舞いの抽出
S108
分析結果記憶部へ格納
選択したプログラムの分析終了

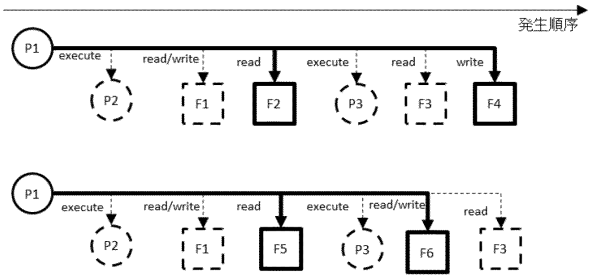
【図９】


発生順序
P2
read/write
read
execute
read
write
F1
F2
P3
F3
F4

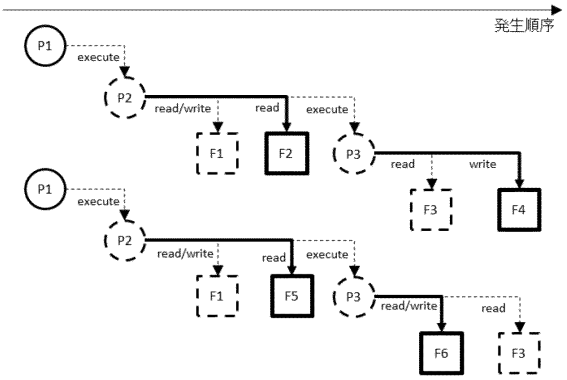
【図１０】


発生順序
P2
read/write
read
execute
read/write
read
F1
F5
P3
F6
F3

【図１１】



【図１２】



【図１３】

(A)

インタプリタの共通の振る舞い 401

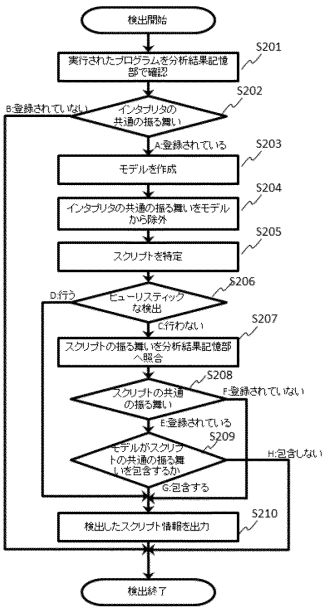
インタプリタ	順序	操作	操作対象
P2	1	read/write	F1
P2	2	execute	P3
P2	3	read	F3

(B)

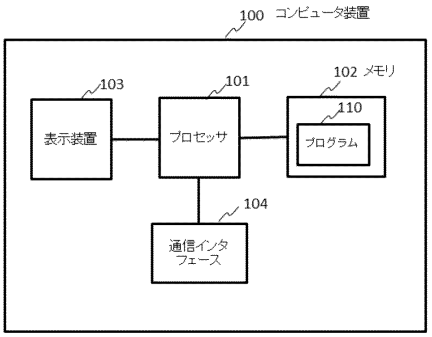
インタプリタとスクリプトの組み合わせにおけるスクリプトの共通の振る舞い 402

インタプリタ	スクリプト	順序	操作	操作対象
P2	F2	1	write	F4
P2	F5	1	read/write	F6

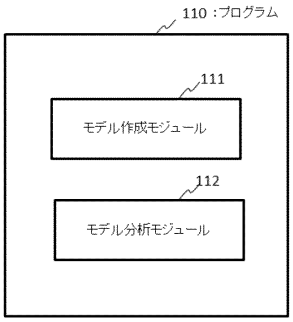
【図１４】



【図１５】


100 コンピュータ装置
103
表示装置
101
プロセッサ
102 メモリ
110
プログラム
104
通信インタフェース

【図１６】


110 :プログラム
111
モデル作成モジュール
112
モデル分析モジュール

フロントページの続き

(56)参考文献　特表２０１７－５２７９３１（ＪＰ，Ａ）
国際公開第２０１７／０１８３７７（WO，Ａ１）
特開２０１７－１２３１４２（ＪＰ，Ａ）

(58)調査した分野　(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　１１／０７
Ｇ０６Ｆ　１１／３４
Ｇ０６Ｆ　２１／５５